# 实验四

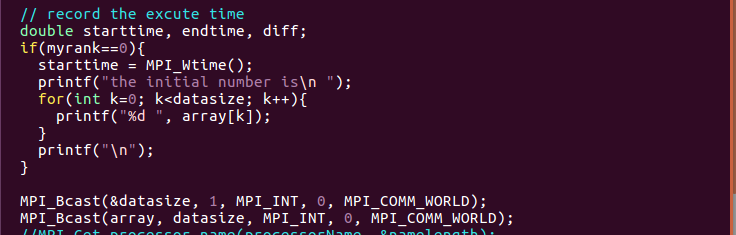
学号：SA17011125

姓名：吴燕晶

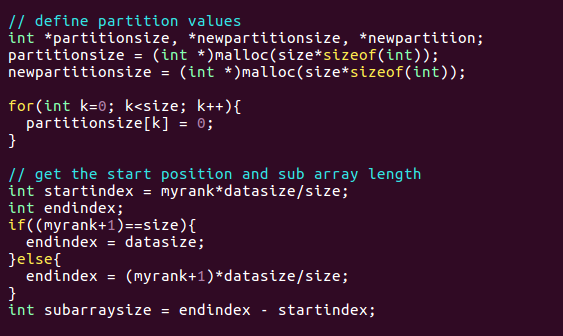
1. 实验题目：利用MPI实现并行排序算法
   1. 实验描述：

并行排序算法(PSRS)的基本原理如下：

1. 均匀划分：将n个元素A[1..n]均匀划分成p段，每个pi处理A[(i-1)n/p+1..in/p
2. 局部排序：pi调用串行排序算法对A[(i-1)n/p+1..in/p]排序
3. 选取样本：pi从其有序子序列A[(i-1)n/p+1..in/p]中选取p个样本元素
4. 样本排序：用一台处理器对p2个样本元素进行串行排序
5. 选择主元：用一台处理器从排好序的样本序列中选取p-1个主元，并播送给其他pi
6. 主元划分： pi按主元将有序段A[(i-1)n/p+1..in/p]划分成p段
7. 全局交换：各处理器将其有序段按段号交换到对应的处理器中
8. 归并排序：各处理器对接收到的元素进行归并排序
   1. 实验要求：
9. 实验结果对待排序数组的大小没有限制，请自行选择适合硬件环境的大小（一般来说，排序的数目在100w以上才能体现出并行的加速情况）。要求在报告中需要有待排序数目为64时的运行结果的截图，以确认排序结果的正确性。
10. 代码中允许使用qsort函数进行对局部数组的排序。
11. 实验环境
    1. 虚拟机：VMware
    2. 操作系统：Ubuntu16.04
    3. 内存：4G
    4. 处理器：4
12. 算法设计与分析
    1. 获取整个数组的起始位置和子数组大小
    2. 调用MPI\_Barrier(MPI\_COMM\_WORLD)进行同步
    3. 调用第一阶段函数，对子数组进行局部排序，以及正则采样
    4. 调用第二阶段函数，对正则采样的样本进行排序，选择主元和主元划分
    5. 调用第三阶段函数，进行全局对全局的发送，并且计算划分的总大小，给新划分分配空间
    6. 调用第四阶段函数，将全局交换后的数据进行归并排序，并发送各排序好的子列表回根进程
    7. 当datasize等于64的时候输出数组
13. 核心代码
    1. 将datasize和数组元素的值进行广播



* 1. 获取整个数组的起始位置和子数组大小



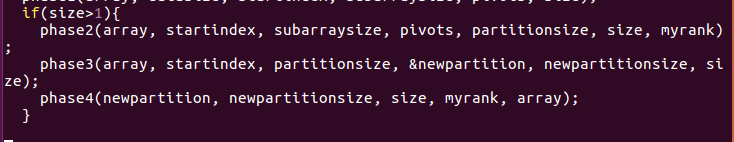
* 1. 调用MPI\_Barrier进行同步



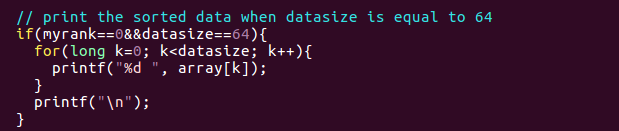
* 1. 调用第一阶段函数



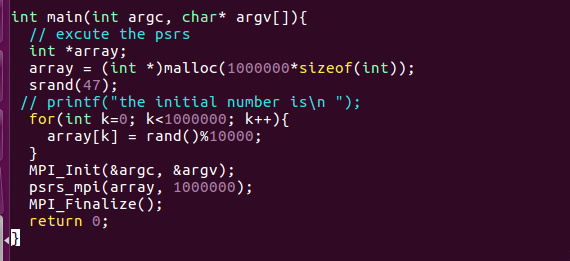
* 1. 调用第二，三，四阶段函数



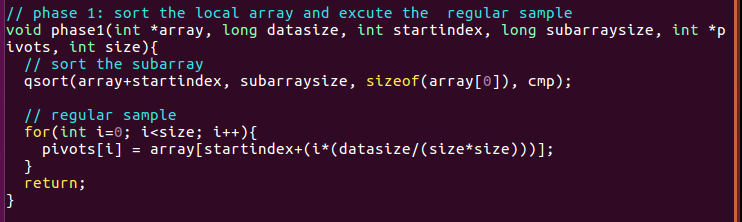
* 1. 输出datasize等于64的时候的数组



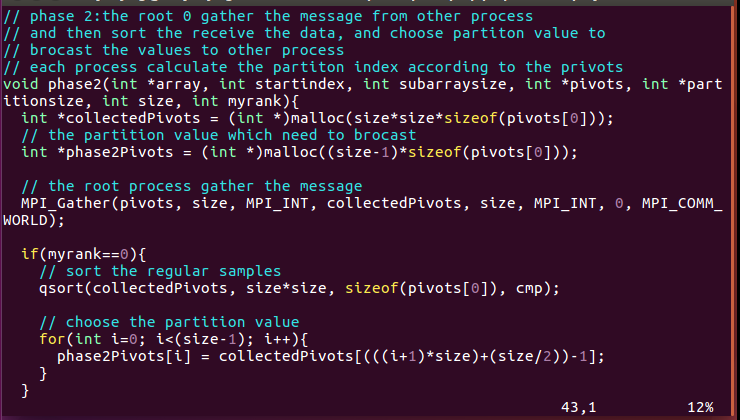
* 1. Main函数代码

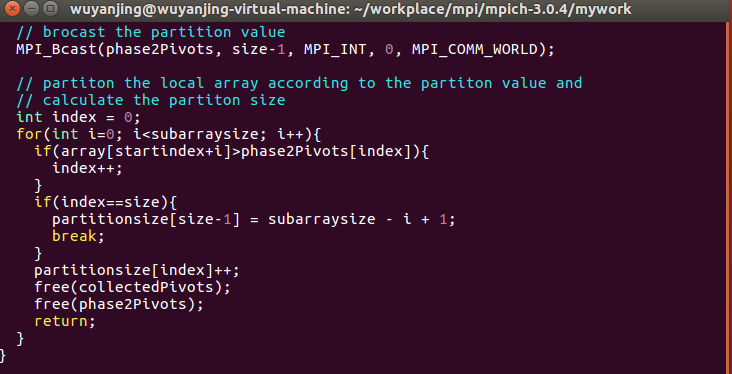


* 1. 具体的四个阶段的函数
     1. 第一阶段

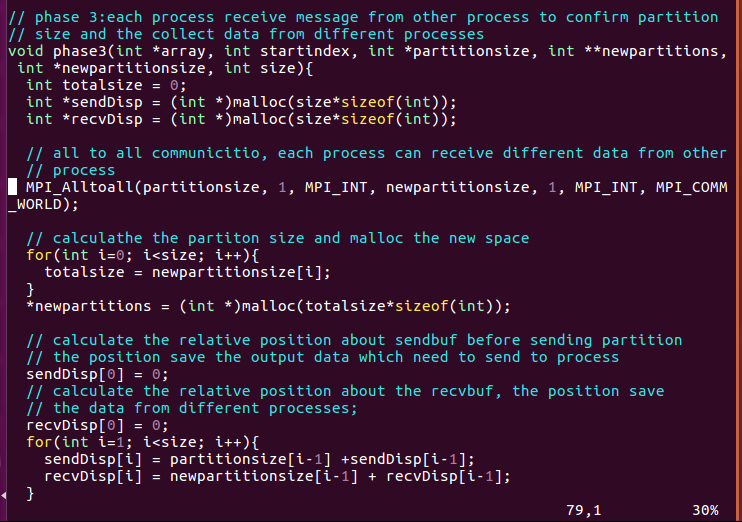


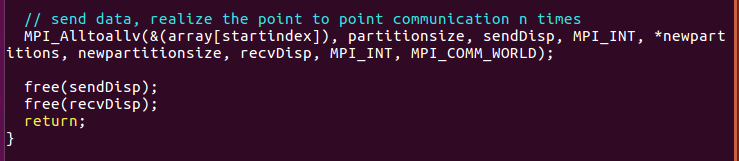
* + 1. 第二阶段



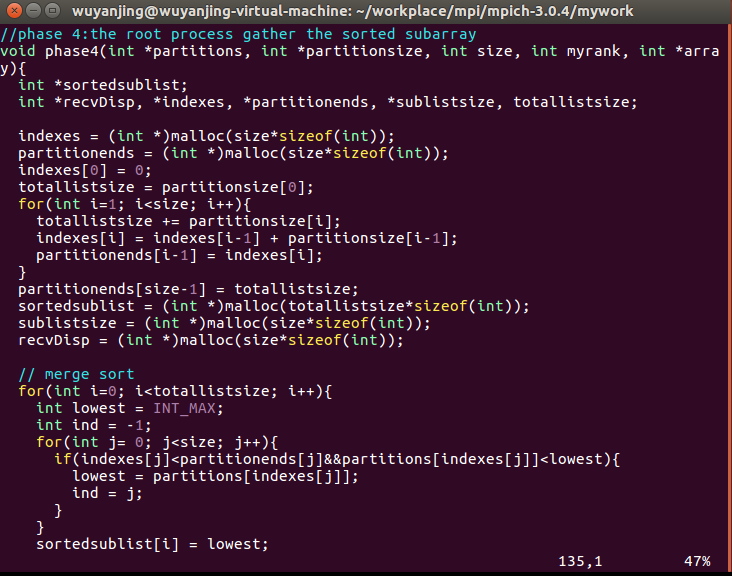


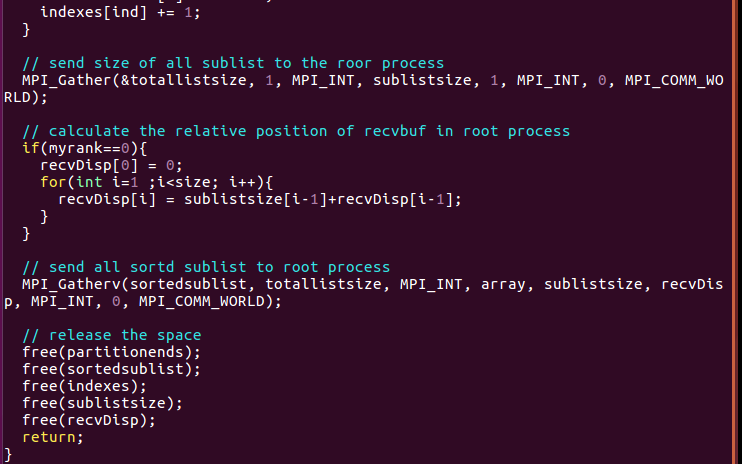
* + 1. 第三阶段



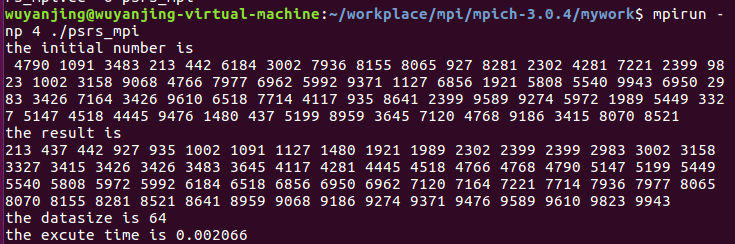


* + 1. 第四阶段





1. 实验结果
   1. 当datasize等于64的时候的结果图：



* 1. 当datasize等于1000000, 5000000, 10000000时的运行时间和加速比

**运行时间**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **规模\进程数** | **1** | **2** | **3** | **4** |
| **N=1000000** | **0.182192** | **0.091883** | **0.062563** | **0.255547** |
| **N=5000000** | **1.143645** | **0.472023** | **0.332487** | **0.264266** |
| **N=10000000** | **2.120692** | **0.999266** | **0.659404** | **0.549921** |

**加速比**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **规模\进程数** | **1** | **2** | **3** | **4** |
| **N=1000000** | **1** | **1.982871** | **2.912128** | **4.475286** |
| **N=5000000** | **1** | **2.422857** | **3.439667** | **4.327636** |
| **N=10000000** | **1** | **2.122250** | **3.216075** | **3.856261** |

1. 分析与总结
   1. 从运行时间表格可以明显看出，随着规模数的增加，运行时间明显增加。对于同一规模来说，无论datasize等于1000000,5000000还是10000000，运行时间都随着进程数的增加而减少。这说明并行化可以通过增加进程数使得并行效果更优。
   2. 从加速比的表格可以明显看出，对于同一规模，随着进程数的进程数的增加，加速比基本呈现线性增加趋势。这说明我们的并行化程序非常的有效。
   3. 我们将串行的程序转为并行化的程序，可以在一定的程度上对程序进行加速，实现效率优化。